**Aim:** To study different word embedding for representation of textual data in vectorized form

**IDE:** Google Colab

# Theory:

[Word Embedding](https://www.geeksforgeeks.org/overview-of-word-embedding-using-embeddings-from-language-models-elmo/) is an approach for representing words and documents. Word Embedding or Word Vector is a numeric vector input that represents a word in a lower-dimensional space. It allows words with similar meanings to have a similar representation.

Word Embeddings are a method of extracting features out of text so that we can input those features into a machine learning model to work with text data. They try to preserve syntactical and semantic information. The methods such as [Bag of Words (BOW)](https://www.geeksforgeeks.org/bag-of-words-bow-model-in-nlp/), [CountVectorizer](https://www.geeksforgeeks.org/using-countvectorizer-to-extracting-features-from-text/) and TFIDF rely on the word count in a sentence but do not save any syntactical or semantic information. In these algorithms, the size of the vector is the number of elements in the vocabulary. We can get a sparse matrix if most of the elements are zero. Large input vectors will mean a huge number of weights which will result in high computation required for training. Word Embeddings give a solution to these problems.

## Need for Word Embedding?

* To reduce dimensionality
* To use a word to predict the words around it.
* Inter-word semantics must be captured.

## How are Word Embeddings used?

* They are used as input to machine learning models. Take the words —-> Give their numeric representation —-> Use in training or inference.
* To represent or visualize any underlying patterns of usage in the corpus that was used to train them.

# Bag of Words

Bag of Words (BOW) is an algorithm that counts how many times a word appears in a document. Those word counts allow us to compare documents and gauge their similarities for applications like search, document classification, and topic modeling.

# Tf-Idf

Tf-Idf is shorthand for term frequency-inverse document frequency. So, two things: term frequency and inverse document frequency. Term frequency (TF) is basically the output of the BoW model. For a specific document, it determines how important a word is by looking at how frequently it appears in the document. Term frequency measures the importance of the word. If a word appears a lot of times, then the word must be important. For example, if our document is “I am a cat lover. I have a cat named Steve. I feed a cat outside my room regularly,” we see that the words with the highest frequency are I, a, and cat. This agrees with our intuition that high term frequency = higher importance.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document)

IDF used to calculate the weight of rare words across all documents. The words that occur rarely in the corpus have a high IDF score. However, it is known that certain terms, such as “I”, “a” may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scaling up the rare ones

IDF(t) = log\_e(Total number of documents / Number of documents with term t in it)

# Pre Lab Exercise:

* 1. Perform BoW vectorization of the corpus
  2. Perform TF vectorization of the corpus

# Program (Code):

To be attached with

1. Perform the TF-IDF vectorization of the corpus

# Results:

To be attached with

# Observation:

**Post Lab Exercise:**

Take three documents and find similarity using

1. BoW vectorization
2. TF Vectorization
3. TF-IDF Vectorization

Comment over the answer obtained in each of the case.